2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

A Memetic Algorithm Using Local Structural Information for Detecting Community Structure in Complex Networks

Caihong Mu, Jin Xie, Ruochen Liu, Licheng Jiao
Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education
International Research Center for Intelligent Perception and Computation
Xidian University, Xi'an, China
E-mail: mucaihongxd@foxmail.com

***Abstract*—Community detection has received a great deal of attention in recent years. Modularity is the most used and best known quality function for measuring the quality of a partition of a network. Based on the optimization of modularity, we proposed a memetic algorithm with a local search operator to detect community structure. The local search operator uses a quality function of local community tightness based on structural similarity. In addition, the tactics of vertex mover is used for reassigning vertices to neighboring communities to improve the partition result. Experiments on real-world networks and computer-generated networks show the effectiveness of our algorithm.**

Keywords—community detection; memetic algorithm; local search; network

I. INTRODUCTION

Many real-world complex systems can be represented as networks, such as the Internet, the world-wide-web, e-mail networks, collaboration networks, social networks and biological networks. In general, these networks consist of a group of vertices and a group of links. One of significant properties of these complex networks is community structure. A community is usually thought of as a group of vertices within which connections are dense while between which they are sparser [1]. Detecting the partitioning of networks has great significance on analyzing the topological structure of the complex networks, understanding the function of complex networks and predicting the behaviors of complex networks.

Numerous methods and algorithms have been proposed for detecting community structure such as graph partitioning, hierarchical clustering, spectral clustering, similarity and dissimilarity measures, heuristic methods and function-based optimization methods [1]. One of the most popular methods is called GN algorithm which is proposed by Newman and Girvan in [2]. The GN algorithm is a divisive hierarchical clustering algorithm which is a process of iterative edges removal from the network.

Newman and Girvan proposed a function called modularity Q which is one of the most popular functions [2]. The community detection problem aims to find a particular clustering with the maximal modularity Q and many algorithms have been proposed based on the measure [4]-[7]. However, maximizing the modularity has been proven to be a nondeterministic polynomial time (NP)-complete problem [8]. Genetic based algorithm is an efficient method for solving NP-complete problem, and able to dramatically reduce the time complexity for solving the problem while ensuring the quality of solutions. In [9] a genetic algorithm was proposed to explore the community structure in social networks by optimizing an objective function. While due to reasons like randomly initial population, unsuitable crossover or mutation operator, the GA based algorithm shows drawbacks including slow convergence and low precision. To overcome these problems, numbers of improved genetic algorithms were proposed. In [7], a Markov random walk based method was used to initial population and a multi-individual crossover operator based on ensemble learning was put forward to replace the traditional crossover operator. Memetic algorithm is a genetic based algorithm combined with a local search strategy. Meme-net, a memetic algorithm with a hill climbing strategy, was proposed to solve the community detection [10]. In addition, some clustering algorithms based on vertex similarity which uses global or local structural information are also effective for detecting community [11]-[13].

In this paper, we propose a memetic algorithm combined with local structural information (MA-LSI) to optimize the modularity. It uses the function of local community tightness to define a local search operator and vertex moving operator. The function of local community tightness proposed in [11] employs the structural similarity. The hill climbing strategy used in [10] is an iterative process that attempts to find a better

This work was supported by the National Basic Research Program (973 Program) of China (No. 2013CB329402), the National Natural Science Foundation of China (Nos. 61003199, 61373111, 61272279, 61072139, 61303032 and 61001202) and the Fundamental Research Funds for the Central Universities (Nos. JB140216, K5051202019, K5051302049).

solution by incrementally changing a single element of the original solution. However, this procedure is time consuming especially when the number of clusters is large, because it needs to calculate fitness function values of all possible neighbor partitions. Compared with the hill climbing strategy used in [10], our local search operator adopts a criterion which makes full use of local structural information of network to generate the neighbor partition of a selected solution. Experimental results show our algorithm is efficient.

II. ALGORITHM DESCRIPTION

A. Problem Definition

Given a network, it can be defined as a graph $G=(V,E)$, where V is the vertex set and E is the edge set. The detection of community structure is to find the division of vertices into groups within which the network connections are dense, while between which they are sparser. Newman and Girvan proposed a function called modularity Q to quantify the quality of a partition of the network [2]. Considering a partition of the network $G=(V,E)$: $G_1(V_1,E_1),\ldots,G_m(V_m,E_m)$, where V_i and E_i are, respectively, the node set and the edge set of G_i for $i=1,\ldots,m$, the well-known modularity Q is given by:

$$Q=\sum_{i=1}^{m}\left[\frac{l_i}{L}-\left(\frac{d_i}{2L}\right)^2\right]. \quad (1)$$

We take the modularity Q as objective function and it should be maximized.

B. Algorithm Framework

Before we give a detailed description of our algorithm, the flow chart is shown in Fig. 1.

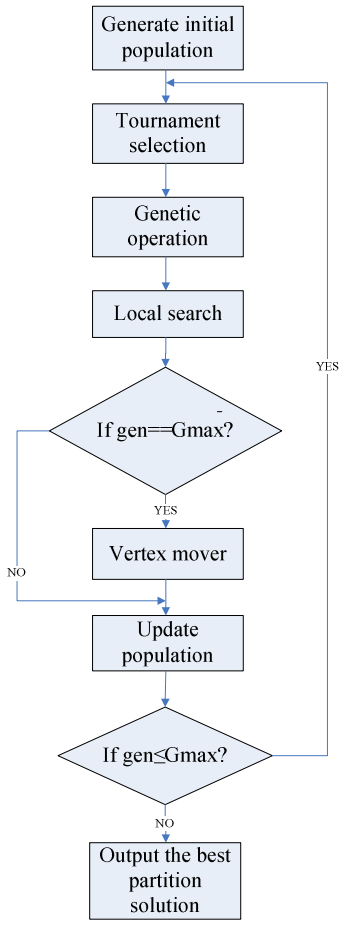

Fig. 1. The flow chart of our algorithm.

Some explanation is required with regard to the flow chart. Gmax and gen represent the maximum number of generations and the current number of iterations respectively. Firstly, the initial population is generated. Then, tournament selection is adopted to generate parental population for mating. The genetic operation includes crossover operator and mutation operator. The local search procedure is executed on the optimal solution in the current generation. The operator of vertex mover is executed only once in the last iteration. When updating the population, the current population and the new generated population are combined to select the top individuals as next generation. When reaching the maximum number of iterations, the algorithm terminates.

Detailed description about the population initialization, selection, genetic operation, local search operator and vertex mover will be given as below.

C. Individuals Encoding and Population Initialization

In this paper, we adopt the direct encoding strategy. A partition of the network G is encoded as an integer string. The length of the integer string equals to the number of vertices in the network and the value of an integer in the string represents the identifier of the community. The vertices having the same identifier are considered in the same community.

We apply the population initialization procedure which has been used in [10]. Firstly, each vertex is put into a different community for all chromosomes, i.e. the initial partition is $\{1,2,\ldots,n\}$, where n is the number of vertices. Then for each chromosome, we randomly select a vertex and assign its community identifier to its neighbors, where the neighbors mean vertices which have links with the selected vertex. This operation is repeated $\alpha\cdot n$ times for each chromosome where α is a model parameter and $\alpha=0.2$ is adopted in our algorithm. The procedure is very fast and results in small communities. Although the result is far away from ideal partition, it can speed up the convergence of our algorithm.

D. Selection and Crossover Operator

Selection operator plays a role of global search in genetic algorithm. Here we choose the deterministic tournament selection.

Crossover operator also plays an important part in global searching and it is the critical operator of genetic algorithm. Considering the special encoding pattern, traditional crossover operators such as single-point crossover or two-point crossover is not suitable. Here we introduce an effective method called

two-way crossover which was proposed in [14]. Given two chromosomes, one of them is taken as source chromosome ch_s and the other as destination chromosome ch_d . Pick a vertex v_p at random; determine its community identifier in the source chromosome and the location of all the vertices belonging to the same community in the source chromosome. Then assign this identifier to the corresponding location in the destination chromosome.

E. Mutation Operator

Here we employ the one-point mutation operator [10]. We pick a vertex randomly on the chromosome, and then the cluster of the vertex is randomly changed to the cluster of one of its neighbors. This operator is repeated n times on the selected chromosome.

F. Local Search Operator

In our algorithm, a new local search operator is employed. The difference between it and the hill climbing strategy is that a criterion is used to determine the neighbor partition which will be described in the following. This local search procedure is employed after genetic operators and only executed on the best individual in the current population. Next, we will illustrate some relevant concepts relating to this operator.

Considering a network which is described as a graph $G=(V,E)$, the structural similarity $s(u,v)$ between two adjacent nodes is given by the following equation

$$s(u,v)=\frac{|\Gamma(u)\cap\Gamma(v)|}{\sqrt{|\Gamma(u)|\cdot|\Gamma(v)|}}. \tag{2}$$

where $\Gamma(u)=\{v\in V\mid\langle u,v\rangle\in E\}\cup\{u\}$, and $\Gamma(u)$ denotes the neighboring nodes of node u . The numerator $|\Gamma(u)\cap\Gamma(v)|$ means the common neighborhood of u and v . Then the tightness T of a local community is then calculated as

$$T_c=\frac{S_{in}^c}{S_{in}^c+S_{out}^c}. \tag{3}$$

where $S_{in}^c=\sum_{i\in c,j\in c}s(i,j)$ is the internal similarity of the community c and $S_{out}^c=\sum_{i\in c,j\notin c}s(i,j)$ is its external similarity. The test whether a node i should join a community c is determined by the value of a criterion. As raised in [11], the criterion to optimize is the tightness gain given by

$$\tau_c(i)=\frac{S_{out}^c}{S_{in}^c}-\frac{S_{out}^i-S_{in}^i}{2S_{in}^i}. \tag{4}$$

Using the theory above, the procedure of finding a neighbor partition based on the function of local community tightness definition is described as follows:

- First, select a vertex randomly on the chosen chromosome.
- Second, find the cluster k which has the community identifier corresponding with the selected vertex. Then determine the neighborhood set N_k of cluster k . Here the neighborhood set N_k means the vertices which have links with the nodes in cluster k .
- Third, determine whether the candidate vertices in the neighborhood set N_k should be included in the community k or not. Here we first select the neighbor vertex with the most possibility a to join in cluster k as the candidate vertex and calculate $\tau_k(a)$ to determine whether it should be added into cluster k or not. The vertex which is most likely to join in cluster means it has the largest similarity with the cluster, i.e., the sum of structural similarities between it and vertices in the cluster is the largest. If $\tau_k(a)>0$, then the vertex a will be inserted into community k . Otherwise, it will be removed from N_k and other vertices will be considered in the descending order of structural similarity.

In order to illustrate the procedure clearly, we consider a simple example of a network with two communities as shown in Fig. 2. The partition can be represented as $\{1,1,1,1,1,2,2,2\}$ and the figures marked on the graph are the structural similarities calculated by (2). The neighborhood set of cluster 1 is $\{6,7\}$ and vertex 6 is more similar to cluster 1 than vertex 7. The value of $\tau_1(6)$ calculated by (4) is 0.1332. Since $\tau_1(6)>0$, we

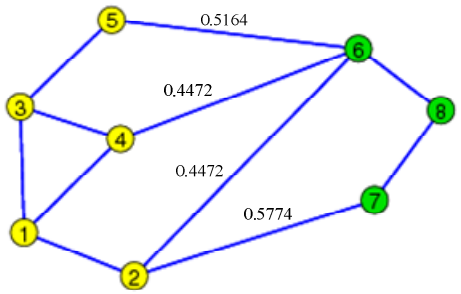

Fig. 2. A simple example of a small network with two clusters.

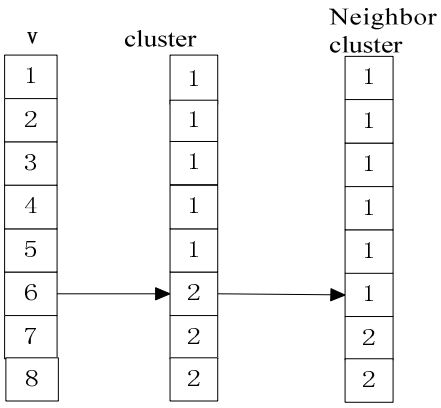

Fig. 3. An illustration of the procedure to generate a neighbor

change the community identifier of vertex 6 into 1 and the neighbor cluster is generated. This process is demonstrated in Fig. 3.

If the process described above produces a better solution, it will be executed iteratively until there are no further improvements. This local search procedure considers the local structural information of the network when finding a neighbor partition of the selected solution, which can not only generate more accurate solution, but also reduce the number of function evaluation.

G. Vertex Mover

The step called vertex mover is applied to further improve the partition result by adjusting vertices between communities. In [12], the VM implementation that vertex v is reassigned from community i to community j is carried out by calculating ΔQ, which is given as follows:

$$\Delta Q = \frac{links(v \leftrightarrow j) - links(v \leftrightarrow i)}{L} - \frac{k_v(d_j - d_{i\setminus v})}{2L^2}. \quad (5)$$

with $links(v \leftrightarrow j)$ the total links between vertex v and community j, k_v the degree of vertex v, d_j the sum degree of all vertices in community j, $d_{i\setminus v} = d_i - k_v$ the corresponding degree for community without vertex. However, upon determining the reassignment of one vertex to any neighboring community means calculating the modularity changes once, and this will increases times of function evaluation. This part is also time-consuming. Consequently, we proposed another strategy to determine vertex moving. The tightness function of a local community which is demonstrated in (3) is still considered in the procedure of vertex mover. Suppose two neighboring communities i and j, the tightness of them can be calculated by (3). If vertex v in community i is reassigned into community j, then the tightness of the two communities is recalculated. The positive increment of tightness results in the reassignment operator. In reality, the process only consider nodes which belong to one community but still have links with other nodes belonging to the neighboring communities.

III. Experiment Results

In this section, we evaluate our algorithm using computer-generated benchmark datasets and some real world datasets. The community structures of them are already known.

We will introduce a similarity measure Normalized Mutual Information (NMI) as described in [15] before we show our experiment result. Considering two partitions A and B of a network in communities, let C be the confusion matrix whose element C_{ij} is the number of nodes of community i of partition A that are also in the community j of the partition B. The normalized mutual information $I(A,B)$ is then defined as

$$I(A,B) = \frac{-2\sum_{i=1}^{c_A}\sum_{j=1}^{c_B} C_{ij}\log(C_{ij}N / C_{i.}C_{.j})}{\sum_{i=1}^{c_A} C_{i.}\log(C_{i.}/N) + \sum_{j=1}^{c_B} C_{.j}\log(C_{.j}/N)}. \quad (6)$$

A. GN Extended Benchmark networks

In this section, the network we adopt here is the benchmark network proposed by Lancichinetti et al. [16], which is an extension of the classic benchmark datasets proposed by Girvan and Newman for testing community detecting algorithms [3]. This extended benchmark network is constituted of 128 nodes divided into 4 communities of 32 nodes each. The average degree of every node is 16 and each node shares a proportion $1-\mu$ of its edges with other nodes of its own community and a proportion μ with other nodes in the network; μ is the mixing parameter. When $\mu < 0.5$, the neighbor nodes of a node inside its community are more than the nodes belonging to the other three communities. As μ increases, community structures of network become more diffused and the detecting algorithms have greater difficulty in mining correct community structures.

We generated 11 computer-generated networks with different value of μ from 0 to 0.5 with interval 0.05 and employed NMI to measure the similarity between the true partitions and the detected ones. For each network, we record the NMI values corresponding with the largest fitness function values over 10 runs. Fig. 4 shows the experimental results. In Fig. 4, y-axis denotes values of NMI, x-axis denotes the mixing parameter μ. To investigate the performance of MA-LSI, this algorithm is compared with Fast Newman (FN) algorithm [4], GA-net algorithm [9] and Meme-net algorithm [10]. Algorithm FN is one of the most classical community detecting algorithms taking modularity Q as objective function. GA-net is the best known algorithm which use genetic algorithm to discover communities in social network by optimizing an efficacious fitness Community Score (CS). Meme-net is proposed by optimizing an objective function called modularity density. In addition, to illustrate the effect of our local search operator, we also test the pure GA version algorithm without the local search operator and vertex mover operator, which is

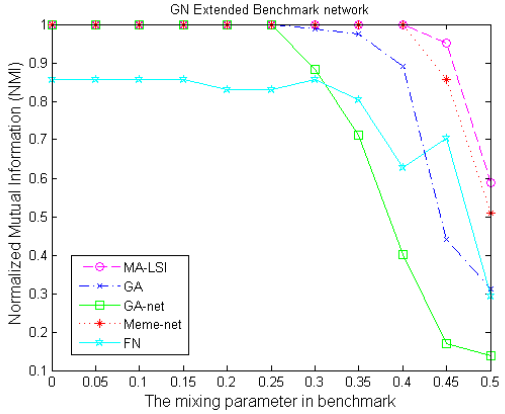

Fig. 4. Max NMI values over 10 runs of different algorithms on GN Extended Benchmark datasets.

named as "GA". Parameters for these four genetic-based algorithms are the same: the population size is 400, the number of iteration is 50, the crossover rate is 0.8 and the mutation rate is 0.2. As we can see from the figure, when $\mu < 0.25$, MA-LSI, GA, GA-net and Meme-net can detect the true partition (NMI=1). When $0.3 \le \mu \le 0.4$, only MA-LSI and Meme-net can detect the true one. While for $\mu = 0.45$, MA-LSI can detect approximately 95% of community structure information. In contrast, the NMI value obtained by Meme-net is about 85%. For $\mu = 0.5$, our algorithm can detect about 58% of the community structure information which is larger than Meme-net. In conclusion, MA-LSI outperforms the other four algorithms in terms of the value of NMI.

B. LFR Benchmark Networks

On those above benchmark networks, all the vertices have approximately the same degree and all communities have the same size. Therefore, new classes of benchmark graphs called LFR have been proposed by Lancichinetti et al. in [16], in which the distributions of node degree and community size are both power laws with tunable exponents ι_1 and ι_2, respectively. Each node shares a proportion $1-\mu$ of its edges with other nodes of its own community and a proportion μ with other nodes in the network; μ is the mixing parameter.

In our experiments, we generated 11 computer-generated networks with different value of μ from 0 to 0.5 with interval 0.05. Each network contains 500 nodes and the cluster size ranges from 10 to 50. $\iota_1 = 2$ and $\iota_2 = 1$, the averaged degree for each node is 20 and the max node degree is 50. The algorithm is run 10 times and the statistical results are showed in Fig. 5.

From the figure we can see that our proposed algorithm performs remarkably well and it is superior to the other three algorithms in terms of the value of NMI. Our algorithm performs well even when $\mu = 0.5$ and it can detected about 91% of the community structure information. As the operation of Meme-net is too time-consuming, the results of Meme-net are not given here.

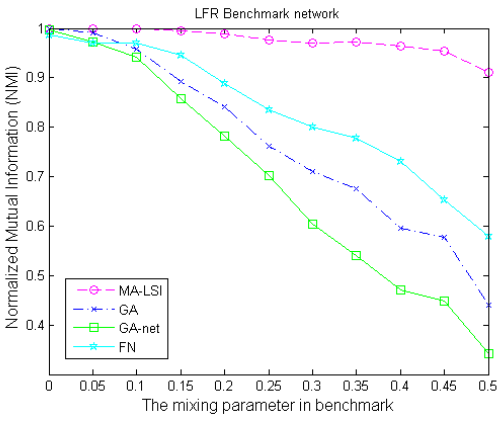

Fig. 5. Max NMI values over 10 runs of different algorithms on LFR Benchmark datasets.

C. Real-world Networks

In this part, we further test our algorithm on data from real-world networks. Here we select four datasets representing which have been widely used as test case for new algorithms.

1) Zachary's karate club: This social network is a well-known graph of friendship between 34 members of a karate club at a US university in the 1970s [17]. As a divergence broke between the administrator of the club and the club's instructor, the club was separated into two parts ultimately.

2) Dolphin social network: This is an undirected social network of frequent associations between 62 dolphins in a community living off Doubtful Sound, New Zealand [18]. The dolphin separated in two groups after a dolphin left the place for some time. Like Zachary's karate club network, the dolphin social network is also used to test algorithms for community detection.

3) American College football: This network is the presentation of the schedule of Division I games for the 2000 season [4]: vertices in the graph represent teams (identified by their college names) and edges represent regular-season games between the two teams they connect. All the teams are divided into 12 conferences containing around 8-12 teams each. Games are more frequent between members of the same conference than those between members of different conference.

4) Books about US politics: This network has been complied by Krebs and Newman [19]. It is a graph of 105 recent books about US politics published around the time of the 2004 presidential election and sold by the online bookseller Amazon.com. Edges between books represent frequent co-purchasing of books the same buyer. Books were divided according to their stated or apparent political alignment, liberal or conservative, except for a small number of books that were explicitly bipartisan or centrist, or had no clear affiliation.

Of all the algorithms we use to be compared with, only algorithm FN take modularity Q as objective function. As for Meme-net and GA-net, the Q values are computed as metric values. We compare the maximum Q values got by our algorithm and other three algorithms. The result is shown in Table I. As we can see, the values obtained by MA-LSI are all larger than that obtained by other algorithms.

Table II records the statistical results over 10 runs for each

TABLE I MAX Q VALUES ON FOUR REAL-WORLD NETWORKS

Algorithm	karate	dolphin	football	polbook
MA-LSI	**0.4198**	**0.5277**	**0.6046**	**0.5272**
FN	0.3807	0.4955	0.5499	0.5020
GA-net	0.4060	0.4923	0.5880	0.5106
Meme-net	0.4161	0.5174	0.6031	0.5255

TABLE II EXPERIMENTAL RESULTS ON FOUR REAL-WORLD NETWORKS

Network	Index	MA-LSI	GA-net	FN	Meme-net
karate	NMImax	**0.6873**	0.6369	0.8372	0.6873
	NMIavg	**0.6873**	0.6369	0.8372	0.6873
dolphin	NMImax	0.6357	0.4266	0.6058	0.6438
	NMIavg	**0.5877**	0.4099	0.6058	0.5690
football	NMImax	**0.9269**	0.9242	0.6538	0.9242
	NMIavg	0.8876	0.9096	0.6538	0.9040
polbook	NMImax	**0.5970**	0.4433	0.5342	0.5901
	NMIavg	0.5704	0.4120	0.5342	0.5866

algorithm on the four small networks whose ground truths are known. We recorded the NMI value corresponding to the max Q and computed the average NMI value over 10 runs. The NMImax values in Table II means the maximum NMI values over the 10 runs. Here we need to state that Meme-net takes modularity density as its fitness function which has a tunable parameter that can help exploring the network at different resolutions. We only recorded the statistical results that the detected number of clusters is same as ours.

For Zachary's karate club, MA-LSI can detect four clusters and the corresponding value of NMI is 0.6873 which is equal to that of Meme-net. FN detects two communities when Q value reaches the maximum, but the maximum Q value is smaller than ours which is shown in Table I. The partition result is displayed in Fig. 6.

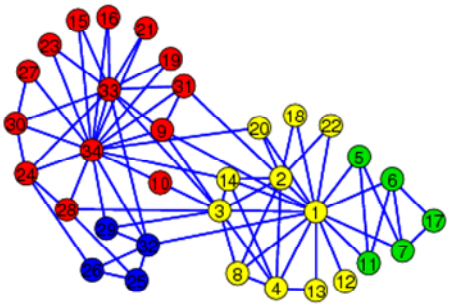

Fig. 6. The clustering result on karate club.

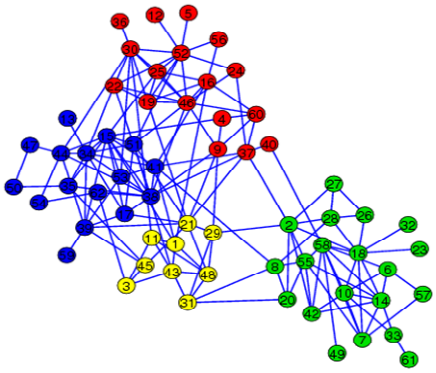

Fig. 7. The clustering result on dolphin network.

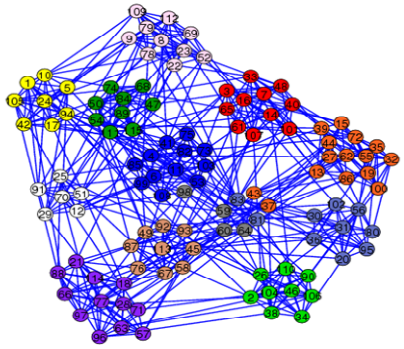

Fig. 8. The clustering result on football network.

For Dolphin social network, the partition corresponding to the NMI value of 0.6357 detected four communities. When Meme-net detects four communities, the maximum NMI value is slightly larger than ours while the average value is smaller. The clustering result is showed in Fig. 7.

For American football team, none of algorithms finds the true partition due to its complicated structure. Our algorithm detects the partitions which own the largest NMI value. Fig. 8 shows clustering results. As we can see, 12 clusters are found and only a few nodes are misplaced.

For Books about US politics, the partition detected by our algorithm owns the maximum NMI value compared with the other three algorithms. And four clusters are found.

D. Comparison of fitness function evaluation times

In this section, the times of fitness function evaluations are compared between MA-LSI and Meme-net. As mentioned before, the population size is 400 and the number of iteration is 50. Each algorithm is executed 10 times and the average number of function evaluations is calculated which is shown in Table III. Here we select different value of μ from 0.25 to 0.5 and the four small real-world networks to test. The average number of MA-LSI is much less.

TABLE III AVERAGE NUMBER OF FUNCTION EVALUATIONS

Algorithm	0.25	0.30	0.35	0.40	0.45
MA-LSI	22649	22863	23035	23053	23063
Mme-net	90072	83729	84541	93087	75774
Algorithm	0.50	karate	dolphin	football	Polbook
MA-LSI	23107	22220	22525	22906	22579
Meme-net	59588	24835	43415	123356	62618

IV. CONCLUSION

In this paper, we proposed a new algorithm MA-LSI to optimize the modularity function Q for community detection. We define a new local search operator and a vertex mover operator using the definition of local community tightness. Local structural information of network is used to guide the local search and good results are obtained with less number of function evaluations. Experiments show its superiority on modularity optimization for community detection problem compared with FN, GA-net, and Meme-net. In the future work, we will extend our algorithm to detect weighed and overlapping networks.

REFERENCES

[1] S. Fortunato, "Community detection in graphs," Physics Reports, pp. 75-174, 2010.

[2] M. E. J. Newman, M, Girvan, "Finding and evaluating community structure in networks," Physical Review E, vol. 69, pp. 026113, Feb. 2004.

[3] M. Girvan, M. E. J. Newman, "Community structure in social and biological networks," Proc. Natl. Acad. Sci. USA, vol. 99, pp. 7821-7826, Dec. 2002.

[4] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," Physical Review E, vol. 69, pp. 066133, Jun. 2004.

[5] O. Gach, J. K. Hao, "A memetic algorithm for community detection in complex networks," Parallel Problem Solving from Nature (PPSN), vol. 7492, pp. 327-336, 2012.

[6] D. Jin, D. He, D. Liu, "Genetic algorithm with local search for community mining in complex networks," in *Conf. 22nd Int. Conf. Tools with Artificial Intelligence*, 2010, pp. 105-112.

[7] D. He, Z. Wang, B. Yang, C. Zhou, "Genetic algorithm with ensemble learning for detecting community structure in complex networks" in *Conf. 2009 IEEE Int. Conf. Computer Science and Convergence Information Technology,* pp. 702-707.

[8] Y. Hu, M. Li, P. Zhang, Y. Fan and Z. Di, "Community detection by signaling on complex networks," Physical Review E, vol. 78, pp. 016115, Jul. 2008.

[9] C. Pizzuti, "GA-Net: A genetic algorithm for community detection in social networks," Parallel Problem Solving from Nature (PPSN), vol. 5199, pp.1081-1090, 2008.

[10] M. Gong, B. Fu, L. Jiao, H. Du, "A memetic algorithm for community detection in social networks," Physical Review E, vol. 84, pp. 056101, 2011.

[11] J. Huang, H. Sun, Y. Liu, Q. Song, T. Weninger, "Towards online multiresolution community detection in large-scale networks," PLoS ONE, vol.6, e23829, Aug. 2011.

[12] P. Schuetz, A. Caflisch, "Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement," Physical Review E, vol. 77, pp. 046112, 2008.

[13] Y. Jiang, C. Jia, J. Yu, "An efficient community detection method based on rank centrality," Physica A, vol. 392, pp. 2182-2194, 2013.

[14] M.Tasgin, A. Herdagdelen, H. Bingol, "Community detection in complex networks using genetic algorithms," ArXiv Condensed Matter e-prints, vol. 4419, pp. 0491, 2007.

[15] L. Danon, A. Dlaz-Guilera, J. Duch, A. Arenas, "Comparing community structure identification," Journal of statistical Mechanics: Theory and Experiment, P09008, 2005.

[16] A. Lancichinetti, S. Fortunato, F. Radicchi, "Benchmark graphs for testing community detection algorithms," Physical Review E, vol. 78, pp. 046110, 2008.

[17] W. W. Zachary, "An information flow model for conflict and fission in small groups," Journal of Anthropological Research, vol. 33, pp. 452-473, Apr. 1977.

[18] D. Lusseau, "The emergent properties of a dolphin social network," Proc. R. Soc. London B (Suppl.), vol. 270, pp. S186-S188, 2003.

[19] M. E. J. Newman, "Modularity and community structure in networks," Proceedings of the National Academy of Science, vol. 103, pp. 8577-8582, 2006.